Information Sciences 662 (2024) 120288

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Disentangling clusters from non-Euclidean data via graph frequency reorganization

Yangli-ao Geng [a,b], Chong-Yung Chi [c], Wenju Sun [a,b], Jing Zhang [d], Qingyong Li [a,b,*]

[a] Key Laboratory of Big Data & Artificial Intelligence in Transportation (Ministry of Education), Beijing Jiaotong University, Beijing, 100044, China
[b] Frontiers Science Center for Smart High-speed Railway System, Beijing Jiaotong University, Beijing, 100044, China
[c] Institute of Communications Engineering, National Tsing Hua University, Hsinchu, 30013, Taiwan
[d] Power Automation Department, China Electric Power Research Institute, Beijing, 100192, China

ARTICLE INFO

Keywords:
Graph filtering
Graph frequency
Graph signal
Frequency decomposition
Unsupervised learning

ABSTRACT

In light of the growing need for non-Euclidean data analysis, graphs have been recognized as an effective tool for characterizing the distribution and correlation of such data, thus inspiring many graph-based developments for various applications such as clustering, of non-Euclidean data. However, under unsupervised scenarios, the construction of graphs from unlabeled data often involves numerous noisy links, consequently leading to serious performance degradation in concerned applications. To resolve this issue, we propose a novel method, referred to as Graph Frequency Reorganization (GFR), to enhance the discriminability of potential clusters and the associated graph quality. GFR shows capability far beyond the suboptimality in unsupervised graph construction. Furthermore, a fast version of GFR is proposed to reduce its computation overhead for large-scale datasets. Consequently, the obtained unsupervised clustering results can be significantly upgraded using the GFR data (i.e., the data after the GFR processing). To evaluate the effectiveness of the GFR, some experimental results on ten real-world datasets are provided to demonstrate that the overall clustering performance of a simple k-means using the GFR data is superior to several state-of-the-art graph-based clustering methods[1].

1. Introduction

Data mining often encounters data with irregular distributions and nonlinear structures [33,14,5]. Graphs utilize vertexes and edges to characterize the distributions and structures, thereupon providing a powerful tool to handle such data. The graph tool has been widely used in many data analytical tasks such as community detection [20], recommendation systems [28] and information retrieval [13].

Unsurprisingly, graphs also play a crucial role in unsupervised learning [18]. Among existing graph-based unsupervised learning methods, the most representative category would be spectral clustering. It models the data as a similarity graph and then transforms

* Corresponding author at: Key Laboratory of Big Data & Artificial Intelligence in Transportation (Ministry of Education), Beijing Jiaotong University, Beijing, 100044, China.
E-mail addresses: gengyla@bjtu.edu.cn (Y.-a. Geng), cychi@ee.nthu.edu.tw (C.-Y. Chi), SunWenJu@bjtu.edu.cn (W. Sun), j_zhang@bjtu.edu.cn (J. Zhang), liqy@bjtu.edu.cn (Q. Li).

[1] The code is available at https://github.com/gyla1993/code_github_230326.

https://doi.org/10.1016/j.ins.2024.120288
Received 24 May 2023; Received in revised form 28 December 2023; Accepted 31 January 2024

Available online 2 February 2024

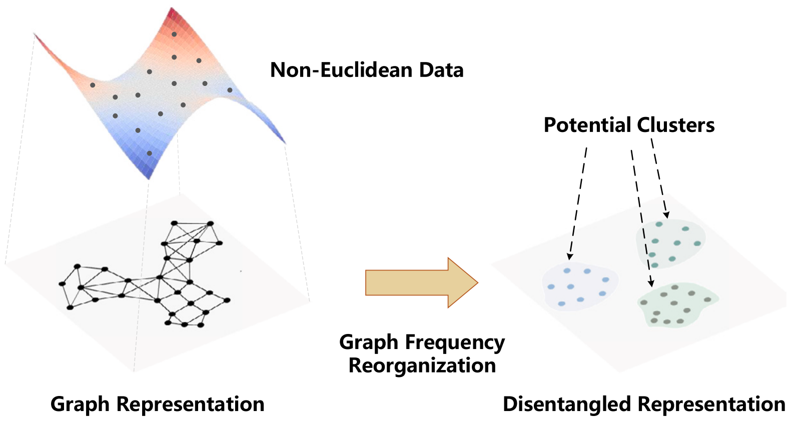

Fig. 1. Illustration for the idea to disentangle clusters from non-Euclidean data via graph frequency reorganization (GFR). Non-Euclidean data can be characterized by a graph. The proposed GFR aims to utilize the graph to acquire a more informative feature representation where potential clusters become disentangled (linearly separable).

the clustering task into a minimal graph-cut problem [31,45]. The representational flexibility of graphs enables spectral clustering methods to handle clusters with irregular shapes. However, these methods rely entirely on the spectrum of the constructed graph, overlooking the rich information embedded in the original data. In contrast, graph filtering methods [33,15,11] use the graph affinity matrix to "filter" graph signals residing on the vertices, yielding the "filtered features" used in the considered application. Graph filtering has achieved great success in many graph-based semi-supervised learning applications [22,41].

Nevertheless, the unsupervised scenario poses a challenge to graph construction due to the absence of data labels. This issue has been the central focus of numerous research endeavors. For instance, Tao et al. [40] introduced an unsupervised cross-domain fault diagnosis method that leverages time-frequency information fusion. Moreover, Zhuang et al. [48,49] devised iterative learning control methods for repetitive systems with random variations in trial lengths. These methods exhibit the effectiveness of iteration-based techniques for repetitive tracking tasks. Such insights have inspired us to employ iterative graph filtering to enhance feature representations for unsupervised learning applications. Existing research [15,2] indicates that multiple operations of graph filtering make the low-frequency components (dependent on the graph quality) critical to the performance of downstream processing. However, in an unsupervised scenario, the graph is constructed from unlabeled data, indicating that few mechanisms exist to guarantee the quality of the constructed graph. As a result, filtering using a low-quality graph may degrade, rather than enhance, the performance of downstream tasks (additional interpretations will be presented in Section 3).

To solve the above problem, we reconsider the fundamental principle of graph filtering from the viewpoint of graph frequency [33,15]. Specifically, we revisit the fundamental frequency concepts of graph signals, leading us to an intriguing perspective on graph filtering. Based on this, we propose an alternative for unsupervised graph filtering, referred to as Graph Frequency Reorganization (GFR). GFR is an iterative approach to alternately improve the discriminability of non-Euclidean data and the quality of the constructed graph, thus enabling it to go beyond the performance sub-optimality suffered by traditional graph filtering. Furthermore, an acceleration strategy is proposed to reduce the computation overhead of GFR for large-scale datasets. The main contributions of this work are summarized as follows:

1. A new frequency reorganization strategy for graph signals is designed, based on which a feature-graph alternating enhancement framework GFR is proposed. GFR offers a new avenue to upgrade the discriminability of potential clusters (as illustrated in Fig. 1) and the quality of the constructed graph.
2. We further develop a fast version of GFR to reduce the computational cost of large-scale data sets, enabling its scalability up to million-level data size.
3. Extensive simulation results and real-data experimental results on unsupervised clustering demonstrate that a GFR-aided clustering (denoted as GFR-C) achieves overall superior performance over several state-of-the-art graph-based clustering methods.

The rest of the paper is organized as follows. Section 2 introduces the related works. Section 3 reviews the basic concepts of graph frequency and presents some insights into traditional graph filtering. Section 4 elaborates on the proposed GFR followed by the fast GFR design. Section 5 provides some simulation and experimental results to demonstrate the efficacy of GFR. Finally, we conclude the paper in Section 6.

2. Related works

Graph Signal Processing. Graph signal processing serves to analyze and extract information from the data that are defined on an irregular discrete domain and represented by graphs [33]. It extends traditional signal processing operations, such as sampling, convolution, and frequency-domain filtering, to signals on graphs [12,7]. Particularly, the graph frequency representation which, from a new perspective in our work, plays an essential role.

Table 1
The upper part lists general mathematical notations and the lower part collectively lists all the graph-related notations.

Notation	Meaning
$\|\cdot\|, \|\cdot\|_F$	ℓ_2-norm of a column vector, Frobenius norm of a matrix
$\mathcal{R}(\cdot), \mathrm{Tr}(\cdot)$	Range space, trace of a matrix
$\odot$	Hadamard (element-wise) product of two homogeneous matrices
$\sqrt{\cdot}$	Element-wise square root of a scalar or a matrix
$\mathbf{X}^\top, \mathbf{x}^\top$	Transpose of matrix $\mathbf{X}$ and vector $\mathbf{x}$
$\mathbf{Diag}(\cdot)$	Diagonal matrix with its diagonal elements given by a column vector
$\mathbb{R}^N, \mathbb{R}^{N\times N}$	Set of N-dimensional vectors, set of $N \times N$-dimensional matrices
$\mathbf{1}_N, \mathbf{I}_N$	All-one vector in $\mathbb{R}^N$, identity matrix in $\mathbb{R}^{N\times N}$
$\mathbb{S}_+^N$	Set of $N \times N$ symmetric matrices with real non-negative elements
$\mathcal{G} = (\mathcal{V}, \mathcal{E})$	Simple undirected graph with vertex set $\mathcal{V}$ and edge set $\mathcal{E}$
$\mathbf{A} = [a_{ij}] \in \mathbb{S}_+^N$	Affinity matrix
$\mathbf{D}$	Degree matrix, i.e., $\mathbf{D} = \mathbf{Diag}(d_1, \dots, d_N)$, where $d_i = \sum_{j=1}^N a_{ij}$
$\hat{\mathbf{A}} \in \mathbb{S}_+^N$	Normalized affinity matrix, i.e., $\hat{\mathbf{A}} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$
$\mathbf{L} \in \mathbb{S}_+^N$	Combinatorial Laplacian matrix, i.e., $\mathbf{L} = \mathbf{D} - \mathbf{A}$
$\hat{\mathbf{L}} \in \mathbb{S}_+^N$	Normalized Laplacian matrix, i.e., $\hat{\mathbf{L}} = \mathbf{D}^{-1/2}\mathbf{L}\mathbf{D}^{-1/2}$

Over-Smoothing and Graph Frequency in Graph Neural Networks (GNNs). The phenomenon of over-smoothing in GNNs refers to the progressive indistinguishability of node features as the depth of the network increases. This effect has captured the attention of researchers, with Li et al. [27] being among the first to document its implications. Over-smoothing is generally believed to adversely affect the performance of downstream tasks, prompting numerous strategies aimed at mitigating its impact [34,6,47]. In an effort to elucidate the mechanics of over-smoothing, Nt et al. [15] established a theoretical connection between GNNs and graph frequency theory. Through this lens, it was discerned that GNNs inherently act as low-pass filters on feature vectors, which may precipitate over-smoothing when the low-frequency components of a graph are misaligned with the requirements of the target task. Bo et al. [2] further explored this concept, demonstrating that both low-frequency and high-frequency signals can be significant in various contexts and should be considered in downstream processes. In advancing the field, Wu et al. [42] introduced the RFA-GNN model, which broadens the scope for adjusting frequency components within GNNs, thereby enhancing the incorporation of high-frequency information. Complementing these frequency-based insights, Bodnar et al. [3] applied cellular sheaf theory to demonstrate a profound connection between graph geometry and both the performance and over-smoothing tendencies of GNNs in heterophilic contexts. In a theoretical stride, Keriven et al. [21] conducted an in-depth analysis of graph smoothing within the realm of simplified GNN frameworks. Extending this line of inquiry, Jiang et al. [19] introduced a sparse-motif ensemble graph convolutional network, designed as an antidote to the over-smoothing challenge. The proposed frequency decomposition technique in our study draws inspiration from these seminal contributions, aiming to further advance the understanding and practical manipulation of graph frequency under unsupervised scenarios.

Graph Spectral Clustering. Spectral clustering (SC) is renowned for its ability to handle non-convex distributed clusters together with its straightforward implementation. In the machine learning community, SC has gained popularity thanks to the works of Shi and Malik [37] and Ng et al. [32]. Subsequently, many variants of SC have been reported [17,36]. Recently, Shi et al. [38] proposed a self-weighting method based on nuclear norm to learn a high-quality similarity graph for multi-view spectral clustering. Sun et al. [39] proposed a lifelong learning framework for spectral clustering via dual memory. One of the drawbacks of SC is its high computational cost. Some researchers have suggested the use of anchors or landmarks to expedite the spectral decomposition in SC [30,4,8]. In addition, there have been other acceleration strategies based on power iteration [29,44]. Different from these works, the proposed GFR adopts a feature-graph alternating enhancement framework. It first constructs a graph from the feature representation. Based on the constructed graph, the feature representation can be enhanced via the proposed frequency reorganization operation. By repeating these two steps, GFR is shown to go beyond the sub-optimality due to unsupervised graph construction, thereby yielding high-quality feature representations.

3. Preliminary

We first present a novel perspective on the basic concepts of graph signals and frequency, laying the foundation for the proposed GFR to be elaborated in Section 4. To facilitate the ensuing discussion, some notations are defined in Tabel 1. Unless stated otherwise, we use italic (e.g., a and A), boldface lowercase (e.g., $\mathbf{a}$) and boldface capital (e.g., $\mathbf{A}$) symbols to denote scalars, vectors and matrices, respectively.

3.1. New perspective on graph frequency

For a given graph $\mathcal{G}$, a graph signal on $\mathcal{G}$ is defined as a vector $\mathbf{z} = [z_1, \dots, z_N]^\top \in \mathbb{R}^N$, where z_i denotes the signal value residing on the vertex i [33]. Let us introduce an alternative definition for the frequency of a temporal discrete signal, which is then used for another perspective on the frequency of a graph signal.

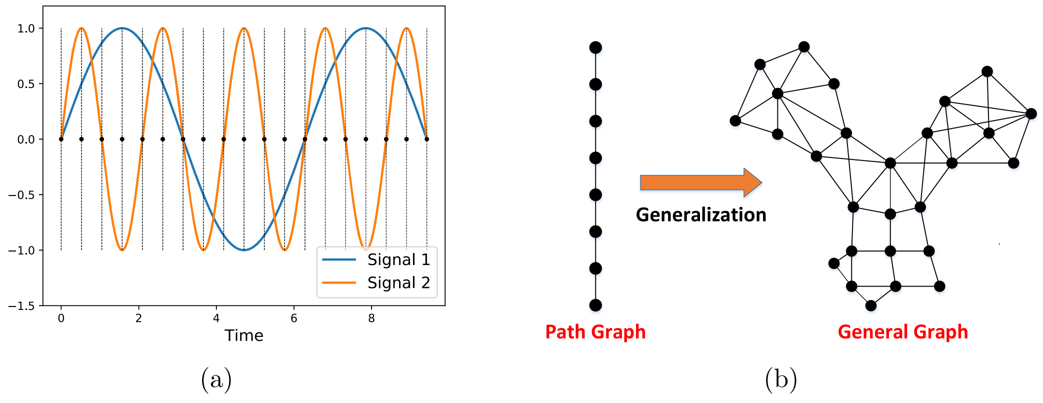

Fig. 2. Illustration of (a) low-frequency temporal signal (signal 1) and high-frequency temporal signal (signal 2), with sampling points indicated by black dots (and hence the sampling interval being the spacing of contiguous sampling points); (b) generalization from a path graph (which corresponds to the underlying structure of the temporal signals in part (a)) to a general graph.

For a discrete temporal signal $\mathbf{z} \in \mathbb{R}^N$, its frequency can be reflected by the changes in signal value between adjacent sampling points, i.e., high frequency implies fast signal value changes. Specifically, these changes between adjacent sampling points can be measured by the squared difference:

$$\mathrm{fre}(\mathbf{z}) = \sum_{i=1}^{N-1} (z_i - z_{i+1})^2, \tag{1}$$

which is referred to as the temporal frequency of $\mathbf{z}$. In the example shown in Fig. 2, the temporal frequency of signal 1 (in blue) and signal 2 (in orange) are approximately 2.4 and 18, respectively, indicating that the former has a lower frequency than the latter, which aligns with intuition. Based on (1), we can naturally extend the frequency of signals over the temporal domain to signals on a graph. As displayed in Fig. 2b, by regarding the sampling points as vertices and the temporal relation of adjacent points as edges, the temporal domain can be characterized by a certain graph: a path graph. Extending the path graph to any general graph, one can generalize the frequency definition (cf. (1)) to any graph. Formally, for a given signal $\mathbf{z} \in \mathbb{R}^N$ on a graph $\mathcal{G}$, its graph frequency is defined as

$$\mathrm{fre}_{\mathcal{G}}(\mathbf{z}) = \sum_{i=1}^{N} \sum_{j>i}^{N} a_{ij}(z_i - z_j)^2, \tag{2}$$

where a_{ij} indicates the affinity between vertexes i and j. Using the Laplacian $\mathbf{L}$ of $\mathcal{G}$, (2) can be further written as

$$\mathrm{fre}_{\mathcal{G}}(\mathbf{z}) = \sum_{i=1}^{N} \sum_{j>i}^{N} a_{ij}(z_i - z_j)^2 = \mathbf{z}^\top \mathbf{L}\mathbf{z}. \tag{3}$$

Intuitively, $\mathrm{fre}_{\mathcal{G}}(\mathbf{z})$ measures the smoothness of the signal $\mathbf{z}$ is on $\mathcal{G}$ [15].

Several studies [31,9] have demonstrated that low-frequency graph signals display a strong correlation with the cluster structure within the graph. However, it is worth noting that any isotropic signal, such as $\mathbf{1}_N$, can achieve the lowest possible graph frequency (zero), providing no valuable information. To circumvent this trivial case, orthogonality constraints are commonly applied to the initial low-frequency components. The mathematical formulation of this problem can be expressed as:

$$\begin{aligned} \min_{\mathbf{z}_1,\dots,\mathbf{z}_C} \quad & \sum_{i=1}^{C} \mathbf{z}_i^\top \mathbf{L}\mathbf{z}_i \\ \text{s.t.} \quad & \mathbf{z}_i^\top \mathbf{z}_i = 1 \\ & \mathbf{z}_i^\top \mathbf{z}_j = 0 \quad (i \neq j), \end{aligned} \tag{4}$$

where C is usually considered as the number of clusters in the dataset. To select suitable low-frequency signals that offer a better balance [31], the normalized Laplacian $\widehat{\mathbf{L}}$ (cf. Table 1) is usually used instead of $\mathbf{L}$ in (4). Furthermore, letting $\mathbf{Z} = [\mathbf{z}_1, \dots, \mathbf{z}_C] \in \mathbb{R}^{N\times C}$, we come up with the following minimization problem for finding the C low-frequency graph signals:

$$\begin{aligned} \min_{\mathbf{Z}\in\mathbb{R}^{N\times C}} \quad & \mathrm{Tr}(\mathbf{Z}^\top \widehat{\mathbf{L}}\mathbf{Z}) \\ \text{s.t.} \quad & \mathbf{Z}^\top \mathbf{Z} = \mathbf{I}_C. \end{aligned} \tag{5}$$

The solution to (5) is known to be the eigenvectors with the C smallest eigenvalues of $\widehat{\mathbf{L}}$ [16], i.e., the eigenvectors with the C largest eigenvalues of $\widehat{\mathbf{A}}$ (cf. Table 1). Specifically, suppose that the eigenvalue decomposition (EVD) of $\widehat{\mathbf{A}}$ is given by

$$\begin{aligned}\widehat{\mathbf{A}} &= \mathbf{P}\boldsymbol{\Lambda}\mathbf{P}^\top = \sum_{i=1}^{N} \lambda_i \mathbf{p}_i \mathbf{p}_i^\top \\ &= \sum_{i=1}^{C} \lambda_i \mathbf{p}_i \mathbf{p}_i^\top + \sum_{i=C+1}^{N} \lambda_i \mathbf{p}_i \mathbf{p}_i^\top \\ &= \underset{\sim}{\mathbf{P}}\underset{\sim}{\boldsymbol{\Lambda}}\underset{\sim}{\mathbf{P}}^\top + \widetilde{\mathbf{P}}\widetilde{\boldsymbol{\Lambda}}\widetilde{\mathbf{P}}^\top,\end{aligned} \tag{6}$$

where $\boldsymbol{\Lambda} = \mathbf{Diag}(\lambda_1, \dots, \lambda_N)$, $\underset{\sim}{\mathbf{P}} = \left[\mathbf{p}_1, \dots, \mathbf{p}_C\right] \in \mathbb{R}^{N \times C}$ collecting the C principal eigenvectors of $\widehat{\mathbf{A}}$ is actually the optimal solution to (5), and $\widetilde{\mathbf{P}} = \left[\mathbf{p}_{C+1}, \dots, \mathbf{p}_N\right] \in \mathbb{R}^{N \times (N-C)}$ gathers the remaining eigenvectors. Note that $\underset{\sim}{\mathbf{P}}$ and $\widetilde{\mathbf{P}}$ respectively form a basis for the low-frequency subspace and the high-frequency subspace, facilitating the orthogonal decomposition of the graph signals of $\mathcal{G}$.

3.2. Graph filtering under a frequency perspective

Graph filtering is an effective tool to handle data analytic tasks with graph structures [22,26]. Specifically, given a data matrix $\mathbf{X} \in \mathbb{R}^{N \times D}$ and a graph $\mathcal{G}$ with the corresponding normalized affinity matrix $\widehat{\mathbf{A}}$, the graph filtering can be expressed as

$$\mathbf{X}_t = \widehat{\mathbf{A}}\mathbf{X}_{t-1},\ t = 1, 2, \dots, \tag{7}$$

where $\mathbf{X}_t$ denotes the data matrix produced by the t-th iteration and $\mathbf{X}_0 = \mathbf{X}$. Substituting (6) into (7) yields

$$\begin{aligned}\mathbf{X}_t &= \widehat{\mathbf{A}}\mathbf{X}_{t-1} \\ &= \underset{\sim}{\mathbf{P}}\underset{\sim}{\boldsymbol{\Lambda}}\underset{\sim}{\mathbf{P}}^\top \mathbf{X}_{t-1} + \widetilde{\mathbf{P}}\widetilde{\boldsymbol{\Lambda}}\widetilde{\mathbf{P}}^\top \mathbf{X}_{t-1} \\ &= \sum_{i=1}^{C} \lambda_i \mathbf{p}_i \mathbf{p}_i^\top \mathbf{X}_{t-1} + \sum_{i=C+1}^{N} \lambda_i \mathbf{p}_i \mathbf{p}_i^\top \mathbf{X}_{t-1}.\end{aligned} \tag{8}$$

It has been a fact that $\widehat{\mathbf{A}}$ guarantees $1 = \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_N \geq -1$ for $1 \leq i \leq N$. Furthermore, when the graph consists of C clusters, by the spectral graph theory [9], we have a good eigenvalue approximation $1 = \lambda_1 \approx \lambda_2 \approx \cdots \approx \lambda_C \gg \lambda_{C+1} \geq \cdots \geq \lambda_N$ and $|\lambda_i| < 1$ $(i = C+1, \dots, N)$ in general. Thus, let $\underset{\sim}{\lambda} = \frac{1}{C}\sum_{i=1}^{C} \lambda_i \approx 1$ and then simplify (8) into

$$\begin{aligned}\mathbf{X}_t &= \sum_{i=1}^{C} \lambda_i \mathbf{p}_i \mathbf{p}_i^\top \mathbf{X}_{t-1} + \sum_{i=C+1}^{N} \lambda_i \mathbf{p}_i \mathbf{p}_i^\top \mathbf{X}_{t-1} \\ &\approx \underset{\sim}{\lambda}\underset{\sim}{\mathbf{P}}\underset{\sim}{\mathbf{P}}^\top \mathbf{X}_{t-1} + \sum_{i=C+1}^{N} \lambda_i \mathbf{p}_i \mathbf{p}_i^\top \mathbf{X}_{t-1}.\end{aligned} \tag{9}$$

After t recursions of (9), we have

$$\mathbf{X}_t \approx \underset{\sim}{\lambda}^t \underset{\sim}{\mathbf{P}}\underset{\sim}{\mathbf{P}}^\top \mathbf{X} + \sum_{i=C+1}^{N} \lambda_i^t \mathbf{p}_i \mathbf{p}_i^\top \mathbf{X} \approx \underset{\sim}{\mathbf{P}}\underset{\sim}{\mathbf{P}}^\top \mathbf{X} \tag{10}$$

for large t due to $\underset{\sim}{\lambda} \approx 1$ and $|\lambda_i| < 1$ $(i = C+1, \dots, N)$. Inspired by (10), we decompose $\mathbf{X}$ into

$$\mathbf{X} = \underbrace{\underset{\sim}{\mathbf{P}}\underset{\sim}{\mathbf{P}}^\top \mathbf{X}}_{\substack{\text{low-frequency} \\ \text{component}}} + \underbrace{\widetilde{\mathbf{P}}\widetilde{\mathbf{P}}^\top \mathbf{X}}_{\substack{\text{high-frequency} \\ \text{component}}}, \tag{11}$$

which is called the *frequency decomposition* of $\mathbf{X}$ with respect to (w.r.t.) the graph $\mathcal{G}$. Note that $\underset{\sim}{\mathbf{P}}\underset{\sim}{\mathbf{P}}^\top$ and $\widetilde{\mathbf{P}}\widetilde{\mathbf{P}}^\top$ act as the projection matrices to the mutual complementary orthogonal subspaces of $\mathcal{R}(\underset{\sim}{\mathbf{P}})$ and $\mathcal{R}(\widetilde{\mathbf{P}})$, respectively. Through the frequency decomposition, one can obtain the low-frequency component $\underset{\sim}{\mathbf{P}}\underset{\sim}{\mathbf{P}}^\top \mathbf{X}$ and the high-frequency component $\widetilde{\mathbf{P}}\widetilde{\mathbf{P}}^\top \mathbf{X}$ of $\mathbf{X}$. Next, let us further discuss the effects of these two components on the unsupervised clustering.

3.3. Further discussion about frequency decomposition

In an ideal case, where the cluster structure is well characterized by the graph, the low-frequency component encapsulates the essential information of discriminative clusters while the high-frequency component mostly comes up with a mixture of miscellaneous structures of different clusters. Under unsupervised scenarios, the constructed graph may involve lots of noisy links, thus blurring the boundaries between different clusters. Consequently, the discriminability of the low-frequency component may also be diminished.

Fig. 3 illustrates the distributions of these two components in both ideal (upper panel) and noisy (lower panel) cases. For the ideal scenario, the low-frequency component $\underset{\sim}{\mathbf{P}}\underset{\sim}{\mathbf{P}}^\top \mathbf{X}$ demonstrates strong discriminability, and the high-frequency component $\widetilde{\mathbf{P}}\widetilde{\mathbf{P}}^\top \mathbf{X}$

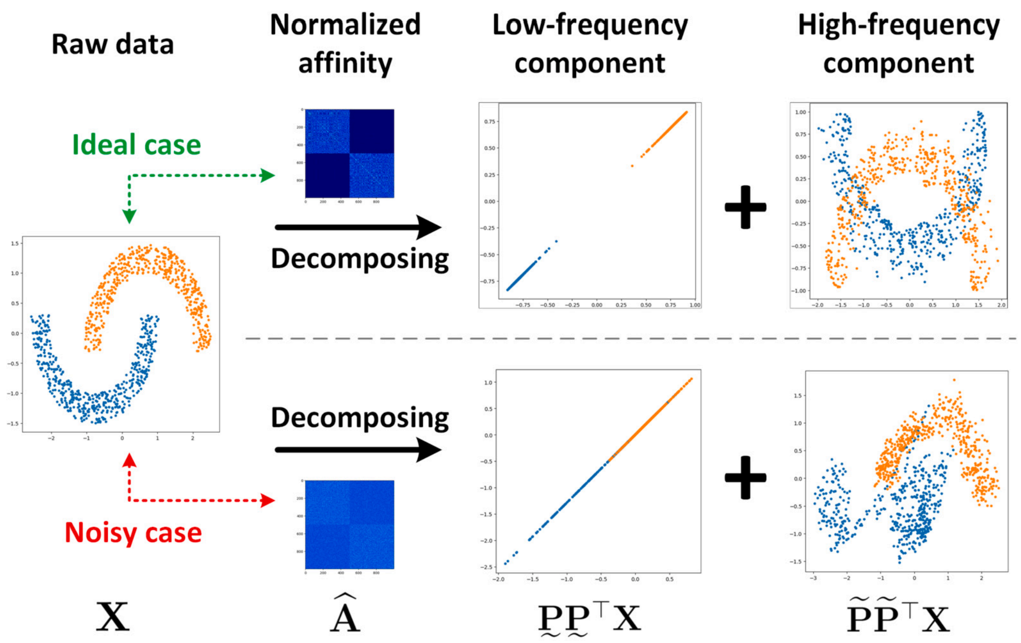

Fig. 3. Illustration of graph projection for the case where the graph specifically characterizes the cluster structure (ideal case, the upper part) and the case with many noisy links (noisy case, the lower part), where the affinity matrix is carefully constructed to match the true class distribution for the former, while the affinity matrix is generated by adding random noises to the data for the latter.

can be viewed as the superposition of centralized distributions of different clusters. On the other hand, in the noisy case, the low-frequency component $\underset{\sim}{\mathbf{P}}\underset{\sim}{\mathbf{P}}^\top\mathbf{X}$ blurs the borders between the two clusters, and in contrast, the high-frequency component $\widetilde{\mathbf{P}}\widetilde{\mathbf{P}}^\top\mathbf{X}$ still contains partial cluster information. The above observations imply that multiple graph filtering operations (cf. (9)) will maintain the low-frequency component outright, thereby yielding the proposed GFR to be presented in the next section.

3.4. Problem formulation

Consider a data matrix $\mathbf{X} \in \mathbb{R}^{N\times D}$, alongside an optional graph $\mathcal{G}$, wherein each node corresponds to a sample within $\mathbf{X}$. Here, N signifies the number of samples, and D represents the data dimension. Our objective is to discover a transformation mapping in an unsupervised scenario:

$$f : (\mathbf{X}, \mathcal{G}) \mapsto (\widehat{\mathbf{X}}, \widehat{\mathcal{G}}), \tag{12}$$

that takes $\mathbf{X}$ and $\mathcal{G}$ as inputs and yields an enhanced data matrix $\widehat{\mathbf{X}} \in \mathbb{R}^{N\times D}$ and a refined graph $\widehat{\mathcal{G}}$. The objective of this mapping is to disentangle potential clusters that are non-Euclidean distributed in $\mathbf{X}$ (meaning they are linearly inseparable), such that they exhibit more pronounced linear separability in $\widehat{\mathbf{X}}$, thereby facilitating a more robust and discernible characterization by $\widehat{\mathcal{G}}$.

4. Methods

This section elaborates on the proposed GFR. We will first detail the proposed GFR in Section 4.1 and then present an acceleration strategy to accelerate its computation, making it more suitable for large-scale datasets, in Section 4.2.

4.1. Graph frequency reorganization

As illustrated in Fig. 4, the proposed GFR consists of three main steps. Given a non-Euclidean dataset (i.e., the raw dataset), GFR first constructs an affinity graph from the raw data. Based on the constructed graph, the low-frequency component and the high-frequency component are acquired through the frequency decomposition presented in Section 3.2. Finally, the re-organized data are constructed with the proportion of the low-frequency component enhanced and that of the high-frequency component reduced. These steps are repeated until a pre-determined stopping rule is met. We will now discuss each step in detail.

Graph construction. As no label information is available, we begin by constructing a graph from the raw data matrix $\mathbf{X} \in \mathbb{R}^{N\times D}$. We first compute the Euclidean pair-distance matrix:

$$\mathbf{E} = \sqrt{(\mathbf{X}\odot\mathbf{X})\mathbf{1}_D\mathbf{1}_N^\top + \mathbf{1}_N\mathbf{1}_D^\top(\mathbf{X}\odot\mathbf{X})^\top - 2\mathbf{X}\mathbf{X}^\top}, \tag{13}$$

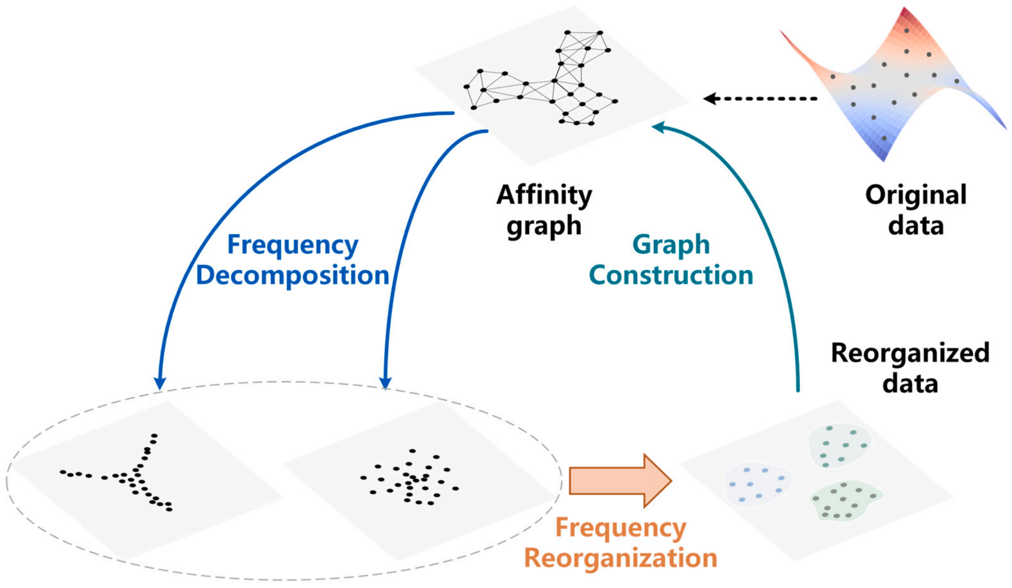

Fig. 4. Synoptic illustration for the three steps of GFR: Graph Construction, Frequency Decomposition and Frequency Reorganization.

where $\mathbf{E} = \left[e_{ij}\right] \in \mathbb{S}_+^N$ and e_{ij} denotes the Euclidean distance between data points $\mathbf{x}_i$ and $\mathbf{x}_j$ (the i-th row and j-th row of $\mathbf{X}$, respectively). We then apply a modified softmax function to $\mathbf{E}$ to obtain an asymmetric similarity matrix $\mathbf{S} = \left[s_{ij}\right] \in \mathbb{R}^{N\times N}$ defined as

$$s_{ij} = \widehat{\sigma}(e_{ij}, K) \triangleq \begin{cases} \dfrac{\exp\{-e_{ij}/\tau\}}{\sqrt{\sum_{k\in\mathcal{N}_K^i}\exp\{-2e_{ik}/\tau\}}} & \text{if } j \in \mathcal{N}_K^i, \\ 0 & \text{otherwise,} \end{cases} \tag{14}$$

where $\mathcal{N}_K^i$ denote the K-nearest-neighbors (KNN) set of $\mathbf{x}_i$ and τ is the temperature parameter, determined by the mean value of the distances between each sample to its K nearest neighbors. We then compute the affinity matrix by $\mathbf{A} = \mathbf{S}\mathbf{S}^\top \in \mathbb{S}_+^N$, which has two key properties: 1) $\mathbf{A}$ is positive semidefinite, i.e., all its eigenvalues are nonnegative; 2) $\mathbf{A}$ is sparse, which can accelerate the subsequent frequency decomposition steps [46].

Frequency decomposition. We normalize the obtained $\mathbf{A}$ by $\widehat{\mathbf{A}} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$, and then obtain the low-frequency component $\underset{\sim}{\mathbf{P}}\underset{\sim}{\mathbf{P}}^\top\mathbf{X}$ and the high-frequency component $\widetilde{\mathbf{P}}\widetilde{\mathbf{P}}^\top\mathbf{X}$ via the frequency decomposition given by (11).

Frequency reorganization. The low-frequency component is enhanced by a factor of $1+\alpha$, while the high-frequency component is reduced by a factor of $1-\alpha$, where $0 \le \alpha \le 1$. They are then combined to form the GFR data:

$$\widehat{\mathbf{X}} = (1+\alpha)\underset{\sim}{\mathbf{P}}\underset{\sim}{\mathbf{P}}^\top\mathbf{X} + (1-\alpha)\widetilde{\mathbf{P}}\widetilde{\mathbf{P}}^\top\mathbf{X}. \tag{15}$$

Obviously, the larger α, the more enhanced the strength of the low-frequency component. For $\alpha = 1$, $\widehat{\mathbf{X}}$ contains only the enhanced low-frequency component, and $\alpha = 0$ means $\widehat{\mathbf{X}} = \mathbf{X}$. The effect of α will be further analyzed through experiments in Section 5.5.

Algorithm 1 summarizes the entire process of the proposed GFR. Among all the rows in Algorithm 1, lines 4, 6 and 9 are the most time-consuming steps, resulting in a complexity of $\mathcal{O}(N^2D + N^3)$. Due to the higher computational complexity of Algorithm 1 for larger data size ($N \times D$), an acceleration strategy is presented in the next section so that it applies to large-scale datasets.

4.2. *Fast graph frequency reorganization*

The substantial computational overhead of GFR arises from the large size of the distance matrix $\mathbf{E} \in \mathbb{R}^{N\times N}$, due to the heavy computations of unnecessary entries. To illustrate, let us consider a simple example. As demonstrated in Fig. 5a, if the distance $\|\mathbf{x}-\mathbf{y}\|$ has already been calculated, the triangle inequality

$$\left|\|\mathbf{x}-\mathbf{y}\| - \|\mathbf{x}-\mathbf{z}\|\right| \le \|\mathbf{y}-\mathbf{z}\|, \tag{16}$$

allows us to infer that $\|\mathbf{x}-\mathbf{y}\| \approx \|\mathbf{x}-\mathbf{z}\|$ when $\|\mathbf{y}-\mathbf{z}\|$ is sufficiently small (i.e., $\mathbf{z}$ is within a small neighborhood of $\mathbf{y}$). This implies that some calculations can be avoided.

Motivated by (16), we introduce the concept of *supporting points*, which can effectively approximate the domain of support for the entire data distribution. Note that similar ideas (known as anchors or landmarks) have been reported to accelerate spectral clustering [30,4,8], but the underlying intuition was not addressed. Specifically, we apply k-means++ [1] to divide all data points into M

Algorithm 1: Graph Frequency Reorganization.

```
Input: Data matrix: $\mathbf{X} \in \mathbb{R}^{N\times D}$, Graph: $\mathcal{G}$ (optional), #KNN: $K$, Low-frequency dimension: $C$, Low-frequency enhancing factor: $\alpha$, #Iterations: $L$
Output: GFR data matrix: $\widehat{\mathbf{X}} \in \mathbb{R}^{N\times D}$
1  if $\mathcal{G}$ is given then
2      $\mathbf{X}_0 = \text{Graph\_Convolution}(\mathbf{X}, \mathcal{G})$;
3  else
4      $\mathbf{X}_0 = \mathbf{X}$;
5  end
6  for $l = 1$ to $L$ do
7      $\mathbf{X}_{l-1} = \mathbf{X}_{l-1} - \frac{1}{N}\mathbf{1}_N\mathbf{1}_N^\top\mathbf{X}_{l-1}$ ;          // Data centered at the origin
8      Calculate $\mathbf{E}$;                                                          // Refer to (13)
9      $\mathbf{S} = \widehat{\sigma}(\mathbf{E}, K)$;                                                    // Refer to (14)
10     $\mathbf{A} = \mathbf{S}\mathbf{S}^\top$;
11     $\mathbf{D} = \mathbf{Diag}(\mathbf{A}\mathbf{1}_N)$;
12     $\widehat{\mathbf{A}} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$;
13     $\widehat{\mathbf{A}} = \underset{\sim}{\mathbf{P}}\underset{\sim}{\mathbf{\Lambda}}\underset{\sim}{\mathbf{P}}^\top + \widetilde{\mathbf{P}}\widetilde{\mathbf{\Lambda}}\widetilde{\mathbf{P}}^\top$;                               // EVD for $\widehat{\mathbf{A}}$, $\underset{\sim}{\mathbf{P}} \in \mathbb{R}^{N\times C}$
14     $\mathbf{X}_l = (1+\alpha)\underset{\sim}{\mathbf{P}}\underset{\sim}{\mathbf{P}}^\top\mathbf{X}_{l-1} + (1-\alpha)\widetilde{\mathbf{P}}\widetilde{\mathbf{P}}^\top\mathbf{X}_{l-1}$;
15 end
16 $\widehat{\mathbf{X}}, \widehat{\mathcal{G}} = \mathbf{X}_L, \widehat{\mathbf{A}}$;
```

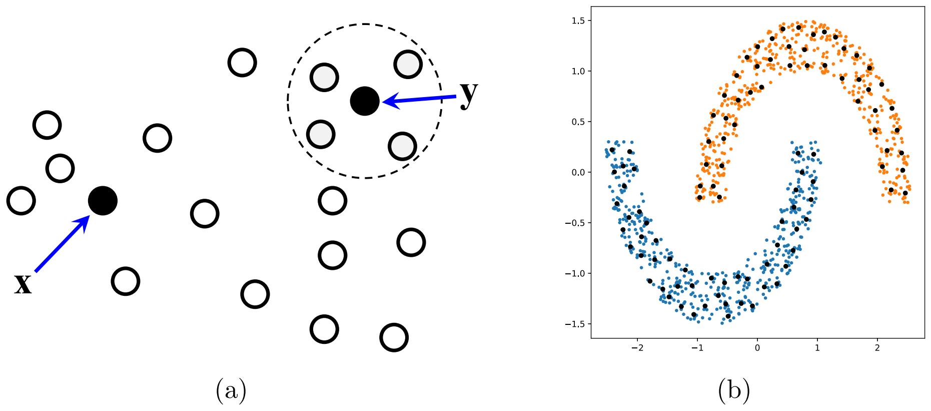

Fig. 5. (a) An illustration for distance computing from a data point $\mathbf{x}$ to all the neighboring data points of $\mathbf{y}$. (b) Visualization for a two-dimensional dataset, where black dots denote the supporting points selected using k-means++.

($\ll N$) fine clusters and employ the corresponding M cluster centers as the supporting points. Thus, a reduced distance matrix $\mathbf{E}_S \in \mathbb{R}^{N\times M}$ between the data points and the supporting points is calculated instead of the full pair-distance matrix $\mathbf{E} \in \mathbb{R}^{N\times N}$.

We then apply the modified softmax function (cf. (14)) to $\mathbf{E}_S$ to derive the corresponding similarity matrix $\mathbf{S}_S$. Following the lines 6, 7 and 8 in Algorithm 1, one can approximate the affinity matrix by $\mathbf{A}_S = \mathbf{S}_S\mathbf{S}_S^\top \in \mathbb{R}^{N\times N}$ and then compute the EVD for the normalized affinity matrix $\widehat{\mathbf{A}}_S = \mathbf{D}_S^{-1/2}\mathbf{A}_S\mathbf{D}_S^{-1/2}$, where $\mathbf{D}_S = \mathbf{Diag}(\mathbf{A}_S\mathbf{1}_N)$. Nevertheless, computing the EVD for $\widehat{\mathbf{A}}_S$ requires computational order $\mathcal{O}(N^3)$, rendering the above supporting point strategy not efficient enough. Fortunately, this high computation cost can be circumvented through an equivalent representation for $\mathbf{D}_S$ and $\widehat{\mathbf{D}}_S$:

$$\widehat{\mathbf{A}}_S = \mathbf{D}_S^{-1/2}\mathbf{S}_S\mathbf{S}_S^\top\mathbf{D}_S^{-1/2} = \widehat{\mathbf{S}}_S\widehat{\mathbf{S}}_S^\top, \tag{17}$$

where $\widehat{\mathbf{S}}_S = \mathbf{D}_S^{-1/2}\mathbf{S}_S$, implying that $\widehat{\mathbf{S}}_S$ can be computed at the cost of $\mathcal{O}(NM)$. Furthermore, the EVD for $\widehat{\mathbf{A}}_S$ can be equivalently performed by conducting singular value decomposition (SVD) on $\widehat{\mathbf{S}}_S$, thereby substantially reducing the computational overhead from $\mathcal{O}(N^3)$ to $\mathcal{O}(NM^2)$.

Algorithm 2 encapsulates all steps of the Fast GFR (FGFR). The two most time-consuming steps are given in line 5 and line 9, which require the complexity of $\mathcal{O}(NMD)$ and $\mathcal{O}(NM^2)$, respectively. Expectantly, the total computational complexity of Algorithm 2 is $\mathcal{O}(NMD + NM^2)$, a significant reduction from $\mathcal{O}(N^2D + N^3)$ (required by Algorithm 1), especially when $M \ll N$.

4.3. Further discussion

Computational Complexity Analysis: A detailed breakdown of the computational complexities for each step in Algorithms 1 (GFR) and 2 (FGFR) is provided in Table 2. For Algorithm 1 (GFR), the most computationally intensive steps involve the construction of the affinity matrix (Line 10) and the frequency decomposition (Line 12). The former involves the multiplication of two $N \times N$

Algorithm 2: Fast Graph Frequency Reorganization.

```
   Input: Data matrix: $\mathbf{X} \in \mathbb{R}^{N \times D}$, Graph: $\mathcal{G}$ (optional), #KNN: $K$, Low-frequency dimension: $C$, Low-frequency enhancing factor: $\alpha$, #Iterations: $L$,
          #Supporting points: $M$
   Output: GFR data matrix: $\widehat{\mathbf{X}} \in \mathbb{R}^{N \times D}$
1  if $\mathcal{G}$ is given then
2      $\mathbf{X}_0 = \text{Graph\_Convolution}(\mathbf{X}, \mathcal{G})$;
3  else
4      $\mathbf{X}_0 = \mathbf{X}$;
5  end
6  for $l = 1$ to $L$ do
7      $\mathbf{X}_{l-1} = \mathbf{X}_{l-1} - \frac{1}{N}\mathbf{1}_N\mathbf{1}_N^\top\mathbf{X}_{l-1}$ ;                 // Data centered at the origin.
8      Select $M$ supporting points $\mathbf{Y} \in \mathbb{R}^{M \times D}$ from $\mathbf{X}_{l-1}$ via k-means++;
9      $\mathbf{E}_S = \sqrt{(\mathbf{X}_{l-1} \odot \mathbf{X}_{l-1})\,\mathbf{1}_D\mathbf{1}_M^\top + \mathbf{1}_N\mathbf{1}_D^\top(\mathbf{Y} \odot \mathbf{Y})^\top - 2\mathbf{X}_{l-1}\mathbf{Y}^\top}$;
10     $\mathbf{S}_S = \widehat{\sigma}(\mathbf{E}_S, K)$;                                                  // Refer to (14)
11     $\mathbf{D}_S = \mathbf{Diag}\left(\bar{\mathbf{S}}_S(\bar{\mathbf{S}}_S^\top\mathbf{1}_N)\right)$;
12     $\widehat{\mathbf{S}}_S = \mathbf{D}_S^{-1/2}\mathbf{S}_S$;
13     $\widehat{\mathbf{S}}_S = \mathbf{U\Sigma V} = \utilde{\mathbf{U}}\utilde{\mathbf{\Sigma}}\utilde{\mathbf{V}} + \widetilde{\mathbf{U}}\widetilde{\mathbf{\Sigma}}\widetilde{\mathbf{V}}$;     // SVD for $\widehat{\mathbf{S}}_S$, $\utilde{\mathbf{U}} \in \mathbb{R}^{N \times C}$
14     $\utilde{\mathbf{X}}_{l-1} = \utilde{\mathbf{U}}(\utilde{\mathbf{U}}^\top\mathbf{X}_{l-1})$;
15     $\widetilde{\mathbf{X}}_{l-1} = \mathbf{X} - \utilde{\mathbf{X}}_{l-1}$;
16     $\mathbf{X}_l = (1+\alpha)\utilde{\mathbf{X}}_{l-1} + (1-\alpha)\widetilde{\mathbf{X}}_{l-1}$;
17 end
18 $\widehat{\mathbf{X}}, \widehat{\mathcal{G}} = \mathbf{X}_L, \widehat{\mathbf{A}}$;
```

Table 2
Computational complexity for each line in Algorithms 1 (GFR) and 2 (FGFR). In practice, we typically have N (#samples) > M (#supproting points) > D (#dimensioans) > C (#clusters). Given this condition, the complexities are listed in ascending order from top to bottom.

Complexity	Algorithm 1 (GFR)	Algorithm 2 (FGFR)
$\mathcal{O}(N \times D)$	Line 7.	Lines 7, 15 and 16.
$\mathcal{O}(N \times M)$	-	Lines 10, 11 and 12.
$\mathcal{O}(N \times D \times C)$	-	Line 14.
$\mathcal{O}(N \times M \times D)$	-	Lines 8 and 9.
$\mathcal{O}(N \times M^2)$	-	Line 13.
$\mathcal{O}(N^2)$	Lines 9, 11 and 13.	-
$\mathcal{O}(N^2 \times D)$	Lines 8 and 14.	-
$\mathcal{O}(N^3)$	Lines 10 and 12.	-
Total	$\mathcal{O}(N^3)$	$\mathcal{O}(N \times M^2)$

matrices (i.e., $\mathbf{A} = \mathbf{S}\mathbf{S}^\top$), and the latter necessitates computing EVD for an $N \times N$ matrix ($\widehat{\mathbf{A}}$). Both of these operations have a computational complexity of $\mathcal{O}(N^3)$.

On the other hand, the bottleneck for Algorithm 2 (FGFR) lies in SVD of an $N \times M$ matrix ($\widehat{\mathbf{S}}_S$), which has a computational complexity of $\mathcal{O}(NM^2)$. Given that N (#sammples) is significantly larger than M (#supporting points), FGFR, with its $\mathcal{O}(NM^2)$ complexity, offers substantially higher efficiency compared to GFR's $\mathcal{O}(N^3)$ complexity. This makes FGFR more scalable when dealing with large-scale datasets.

It is noteworthy that the majority of the computational steps in FGFR are matrix operations, which are particularly amenable to acceleration using a GPU-based parallel computation framework, such as pyTorch.[2] This further enhances the scalability and efficiency of FGFR.

Improvements in Methodology: The major advancements of GFR over established methods should be owing to two aspects: the feature-graph alternating enhancement framework and the frequency reorganization strategy. The former offers a new avenue to mitigate the sub-optimality arising from graph construction using unlabeled data, and the latter provides a specific recipe for how to enhance the feature presentation through a constructed graph. Specifically, GFR employs an iterative method, diverging from conventional approaches that seek to directly construct a high-quality graph in the raw feature space. The process starts by building an initial graph from the original feature representation, which is then improved through the proposed frequency reorganization strategy. With the refined feature representation, the prospects of constructing a superior-quality graph are heightened, thereby enabling us to achieve superior feature representations. Repeating these steps, we eventually obtain high-quality feature representations and corresponding graph structures. Through these innovative designs, the proposed GFR provides a promising solution to the

[2] https://pytorch.org/.

Table 3
Datasets used to evaluate the performance of the proposed GFR.

Dataset	#Samples	#Clusters	Extractor	#Dimension
Gaussian4	2,000	4	None	2
Tetris	8,000	6	None	2
Iris	150	3	None	4
Seed	219	3	None	7
Wine	178	3	None	13
OrlFace10 [35]	100	10	None	92×112
OrlFace40 [35]	400	40	None	92×112
USPS [25]	9,298	10	CNN_AE	8
FMNIST [43]	70,000	10	CNN_AE	8
MNIST [24]	70,000	10	CNN_AE	8
STL10 [10]	13,000	10	ResNet18	512
CIFAR10 [23]	60,000	10	ResNet18	512

sub-optimality arising from graph construction using unlabeled data and bolsters data representations for downstream tasks, thereby establishing the novelty and contributions of our work.

In addition, compared to traditional clustering algorithms, the proposed GFR exhibits broader usability and better interpretability. Traditional clustering algorithms are mostly designed to directly assign cluster labels, denoted as $\mathbf{Y} \in \mathbb{R}^{N\times 1}$, to each sample within the original data matrix $\mathbf{X} \in \mathbb{R}^{N\times D}$. While $\mathbf{Y} \in \mathbb{R}^{N\times 1}$ effectively indicates which samples likely originate from the same category, it omits the distribution information inherent in the raw data space. This omission incurs the loss of valuable information for downstream tasks. In contrast, our proposed methods, GFR and FGFR, aspire to produce an improved data representation, $\widehat{\mathbf{X}}$. This representation not only encapsulates the vital distribution information from the original data but also bolsters the discriminability of potential clusters. As a result, $\widehat{\mathbf{X}}$ emerges as a more potent representation for facilitating downstream analyses. Moreover, GFR and FGFR provide a transparent trial (a two-dimensional example is shown in Fig. 6) of the transformation process from $\mathbf{X}$ to $\widehat{\mathbf{X}}$, enabling users to gain deeper insights into the nuances of the feature transformation.

5. Experiments

5.1. Experimental setup

We conduct experiments on a 64-bit Ubuntu 20.04.4 computer with 14 Intel E5-2680 2.40GHz CPUs, 256 GB memory, and 8 NVIDIA RTX 2080 Ti GPUs. The code is available at https://github.com/gyla1993/code_github_230326.

5.1.1. Datasets

The datasets used in the experiment are listed in Table 3 together with brief directions for each dataset. The first two (Gaussian4 and Tetris) are simulation datasets to be used to show distinctions in the data distribution between the proposed GFR and a commonly used graph iteration. The others are real datasets (including 5 small-size ones and 5 large-size ones). Detailed descriptions about these datasets can be found in the cited references or websites.

5.1.2. Baseline methods and parameter settings

To evaluate the performance of the proposed GFR, a GFR-aided unsupervised clustering algorithm, denoted as GFR-C, is considered. Specifically, we process the raw data with GFR (i.e., applying Algorithm 1 and Algorithm 2 to the five small and the five large datasets, respectively) to obtain the GFR data for each dataset, and then performing k-means++ (KM++) [1] directly on the GFR data to obtain clusters. For performance comparison, we select four existing graph-based benchmark clustering methods as baselines, including spectral clustering (SC) [31], low-frequency clustering (LFC), power iteration clustering (PIC) [29] and graph filtering (GF). All four baseline methods are tested through the common procedure: a) extracting features from the raw data by its own graph-based strategy and b) applying k-means++ (KM++) [1] to the extracted features to yield clustering results. Step b) is identical for all of them. Their feature extractions in step a) are addressed respectively, as follows. After constructing an affinity graph from the raw data, SC employs the first several eigenvectors of the associated affinity matrix as its features; LFC adopts the low-frequency component of the constructed graph as its features; PIC extracts features by performing several graph iterations on an N-dimensional normalized vector with the initial value for each entry proportional to the degree of the associated vertex; GF extracts features by performing graph iterations on the raw data matrix.

All the methods (including the proposed GFR) under test share the same graph construction process (i.e., construction of the affinity matrix $\mathbf{A}$ from the raw data matrix $\mathbf{X}$), elaborated in the second paragraph of Section 4.1, where K (the number of nearest neighbors) is taken from the set $\{4, 8, 16, 32\}$. Moreover, GFR has an additional parameter $\alpha \in \{0.025, 0.05, 0.075, 0.1\}$. For the three iteration-based methods, PIC, GF and GFR, their iteration numbers are respectively set to 120, 30 and 30, where the iteration number of PIC is much larger than the other two due to its slow convergence rate. When applied to the large-scale datasets, all the graph-based methods utilize the supporting point strategy described in Section 4.2, where M (the number of supporting points) is set to 500. The low-frequency dimension C is set to the true number of clusters, which is the prior knowledge in our experiments. The best performance is obtained for all the methods under test by searching through the aforementioned parameter spaces.

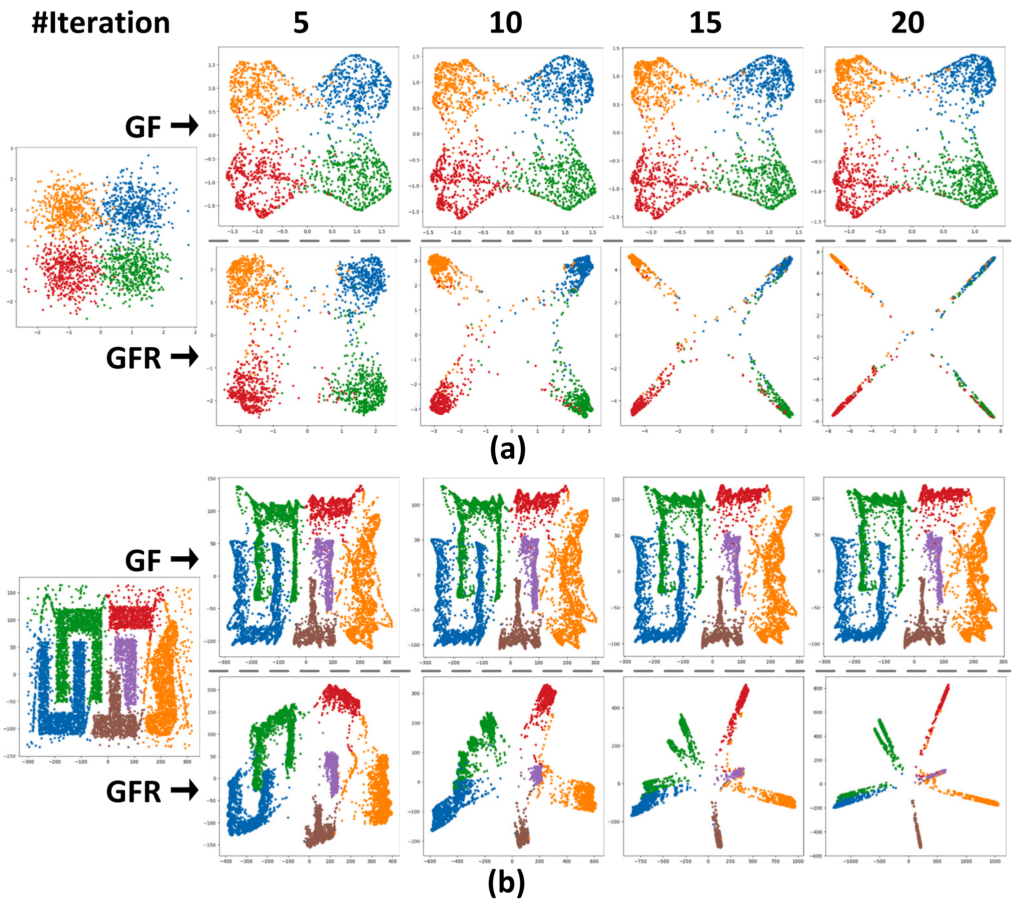

Fig. 6. The visualization of different iterations between GF (the top row in each sub-figure) and the proposed GFR (the bottom row in each sub-figure) on the two simulation datasets: (a) Gaussian4 and (b) Tetris. The generated features in #iteration 5, 10, 15 and 20 are visualized.

5.1.3. Performance metrics

We evaluate the clustering quality of all the algorithms under test by two classic measures: adjusted rand index (ARI) and normalized mutual information (NMI). The larger the two measures, the better the clustering quality. Both ARI and NMI range between 0 and 1.

5.2. Visual results on simulation datasets

By visualizing different iterations of GF and the proposed GFR on two simulation datasets Gaussian4 and Tetris, their different iteration characteristics can be illustrated.

As shown in Fig. 6, Gaussian4 consists of four partial-overlapping Gaussian clusters. We observe that GF just shrinks the distribution of each cluster. GFR not only shrinks the distribution of each cluster but also unscrambles the boundaries between different clusters. Note that we re-scale the data after each iteration to focus on their distribution characteristics. Fig. 6b illustrates the results of Tetris, where six irregular clusters are tangled seriously. For GF, the similar behaviors shown in Fig. 6a can also be observed in Fig. 6b: it only compresses the area of each cluster, without changing their distribution characteristics (linear inseparability). In contrast, as shown in the bottom row of Fig. 6b, GFR gradually disentangles these irregular clusters and unfolds them.

5.3. Quantitative results

Quantitative results (ARI and NMI) of all the six methods on the ten datasets are given in Table 4.

From the results shown in Table 4, we have some observations as follows. Obviously, KM++ performs better than PIC, which performs worst in terms of both ARI and NMI. The SC method, one of the most classic graph-based methods, demonstrates significant improvements over KM++ on several datasets such as OrFace40, USPS and MNIST. LFC performs better than SC, especially on Iris and OrFace40, though their overall performances are competing with each other. Moreover, GF displays better overall performance than both SC and LFC, perhaps because it can capture more subtle clustering information through the attenuation of the high-frequency component in the ensuing graph iterations. Nonetheless, its performance is still limited by the capability of the initial

Table 4
Quantitative results (ARI and NMI) of various clustering methods on the ten datasets. The best (second) performance for each dataset is highlighted in boldface (underline).

Method	Iris		Seed		Wine		OrlFace10		OrlFace40		USPS		MNIST		FMNIST		STL10		CIFAR10	
	ARI	NMI	ARI	NMI	ARI	NMI	ARI	NMI	ARI	NMI	ARI	NMI	ARI	NMI	ARI	NMI	ARI	NMI	ARI	NMI
KM++	.730	.758	.797	.754	.898	.876	.913	.949	.584	.847	.673	.744	.761	.791	.498	.664	.794	.823	.270	.386
SC	.772	.786	.811	.761	**.915**	**.893**	.954	.971	.692	.902	.760	.838	.893	.887	.492	.686	.740	.808	.274	.446
LFC	.818	.804	.811	.761	**.915**	**.893**	.954	.971	.724	.912	.764	.840	.891	.887	.492	.689	.747	.807	.277	.433
PIC	.674	.759	.288	.342	.823	.805	.643	.830	.305	.692	.468	.579	.446	.604	.405	.587	.516	.648	.070	.139
GF	.834	.817	.836	.785	**.915**	**.893**	.954	.971	.722	.912	.748	.822	.892	.887	**.505**	.682	**.811**	.834	.311	.434
GFR-C	**.886**	**.862**	**.837**	**.787**	**.915**	**.893**	**.977**	**.986**	**.753**	**.927**	**.768**	**.850**	**.906**	**.899**	.503	**.695**	**.811**	**.842**	**.352**	**.483**

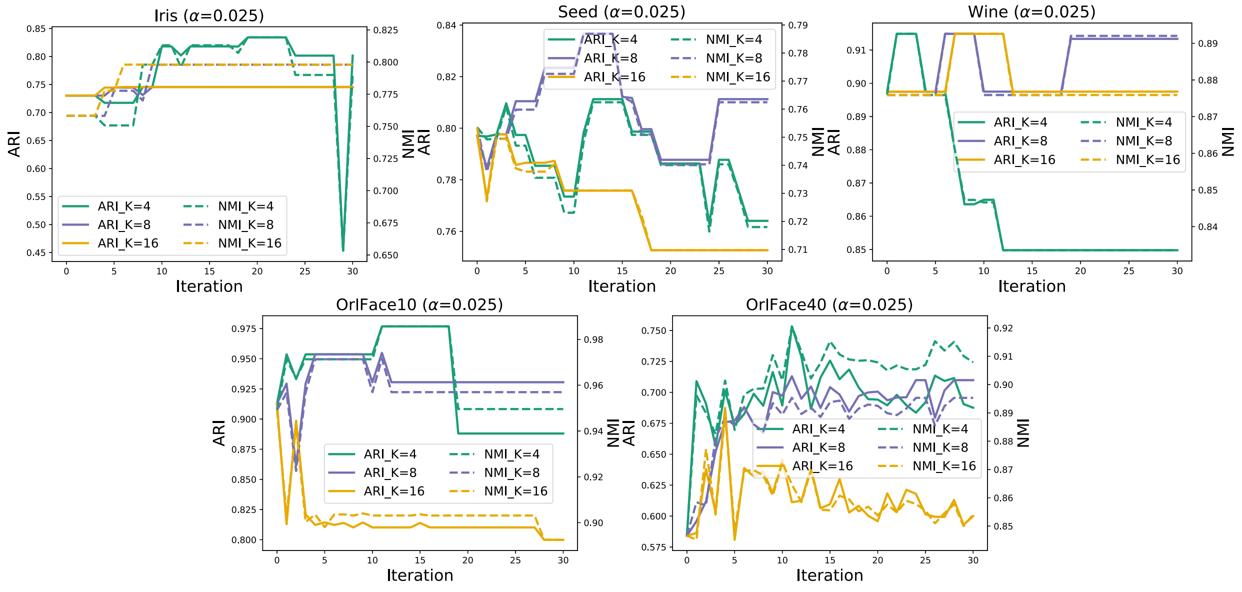

Fig. 7. GFR-C performance (measured by ARI and NMI) versus iteration number on the five small datasets for $K \in \{4, 8, 16\}$ and $\alpha = 0.025$.

constructed graph. Finally, GFR-C achieves the best performance for almost all datasets, demonstrating that the proposed GFR can go beyond the limitation due to imperfect graph construction. Note that SC, LFC, GF and GFR-C obtain the same scores on Wine, indicating that the initially constructed graph is also quite close to the one obtained by GFR for this dataset, thus yielding almost the same good performance. Therefore, these experimental results have justified that the graph frequency representation and filtering can better characterize data manifold, thus enhancing the subsequent clustering quality.

5.4. Parameter analysis on K

As a key parameter of GFR, the number of neighbors K controls the degree of connectivity of the constructed graph. An excessive K blurs the boundaries between clusters while a too-small K reduces the connectivity within a cluster. It is needed to choose a proper K to balance the separability between clusters and compactness within each cluster for different datasets. To this end, we fix the low-frequency enhancing factor $\alpha = 0.025$ and investigate the effect of GFR on the clustering performance (in terms of ARI and NMI) as a function of the iteration number under different choices of $K \in \{4, 8, 16\}$. The corresponding results for $K = 32$ are omitted since they are similar or inferior to those for $K = 16$.

Fig. 7 illustrates the performance curves for the five small datasets. It can be observed from this figure that, although $K = 4$ and $K = 8$ are better choices than $K = 16$ for most datasets, there is not a common rule for different datasets, perhaps due to large difference in cluster sample sizes (cf. Table 3). On the other hand, Fig. 8 displays the performance curves for the five large datasets with $M = 500$ (the number of supporting points). It can be seen that the clustering performance for $K = 4$ is the best for all the five datasets except the STL10 dataset, for which $K = 8$ achieves the most preferable clustering performance. Overall, for small datasets careful tuning for K is needed, while for large datasets a small K (e.g., 4 or 8) is suggested when the supporting point strategy is utilized.

5.5. Parameter analysis on α

To show the effect of the low-frequency component enhancing factor α of the proposed GFR on the clustering performance (in terms of NMI), we show the 2-dimensional heatmap (distribution) of NMI versus α and the iteration number in Fig. 9 for all the ten datasets.

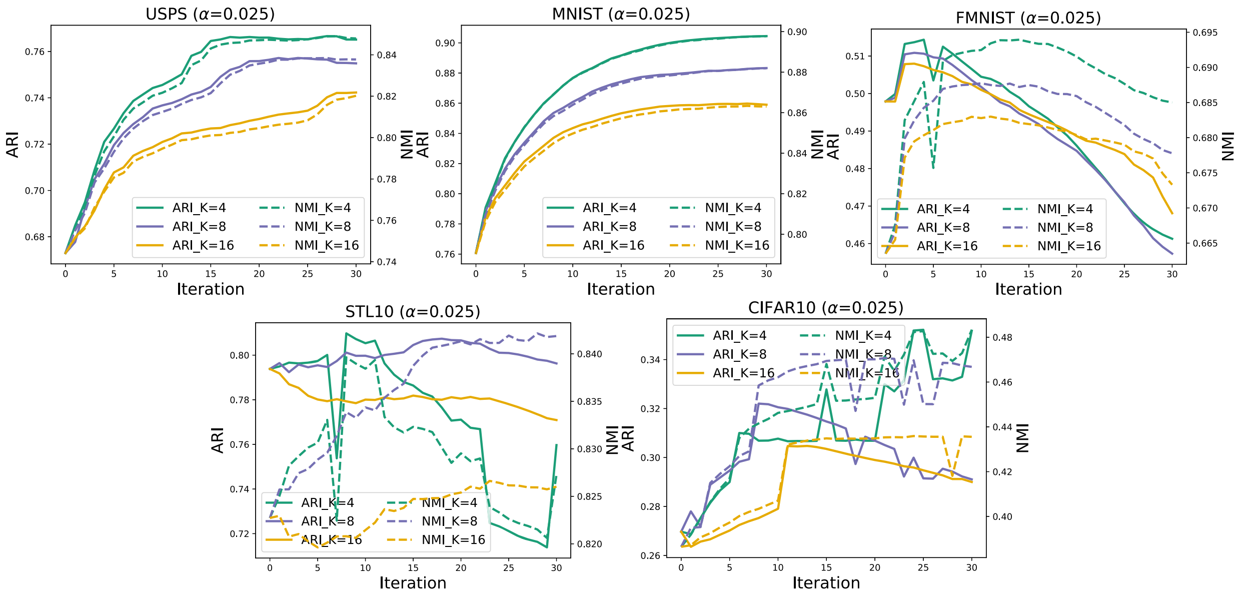

Fig. 8. GFR-C performance (measured by ARI and NMI) versus iteration number on the five large datasets for $K \in \{4, 8, 16\}$, $\alpha = 0.025$ and $M = 500$.

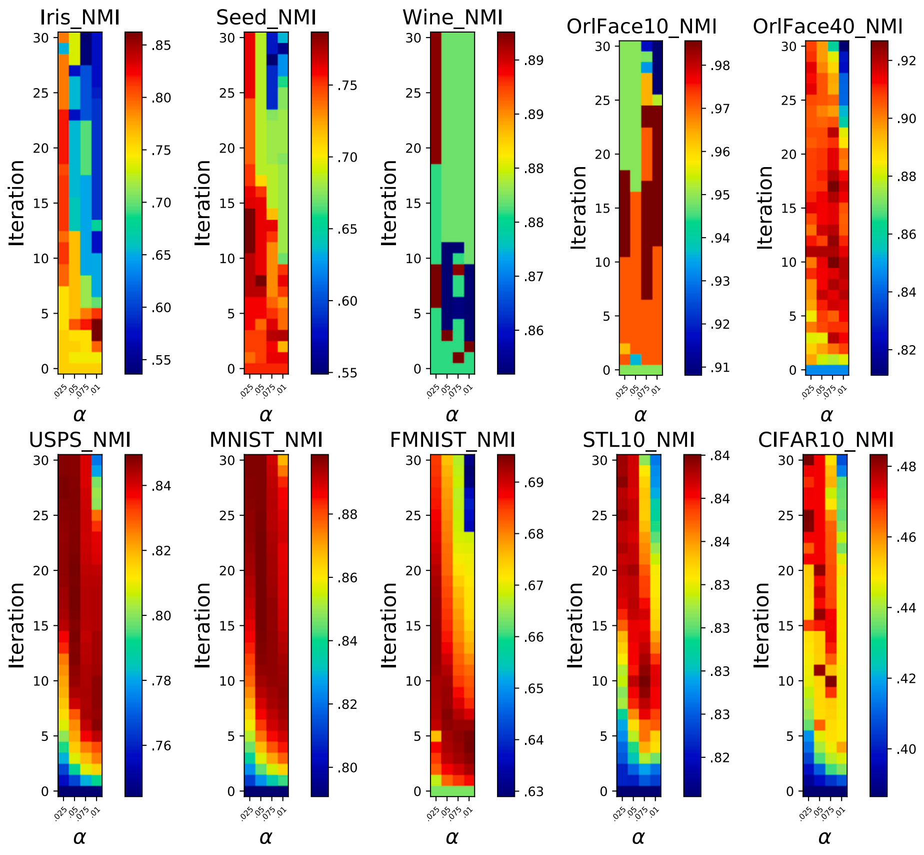

Fig. 9. The heatmap of GFR-C performance (measured by NMI) versus the low-frequency enhancing factor α and iteration number, for all the datasets, where a proper $K \in \{4, 8, 16, 32\}$ is used for each dataset.

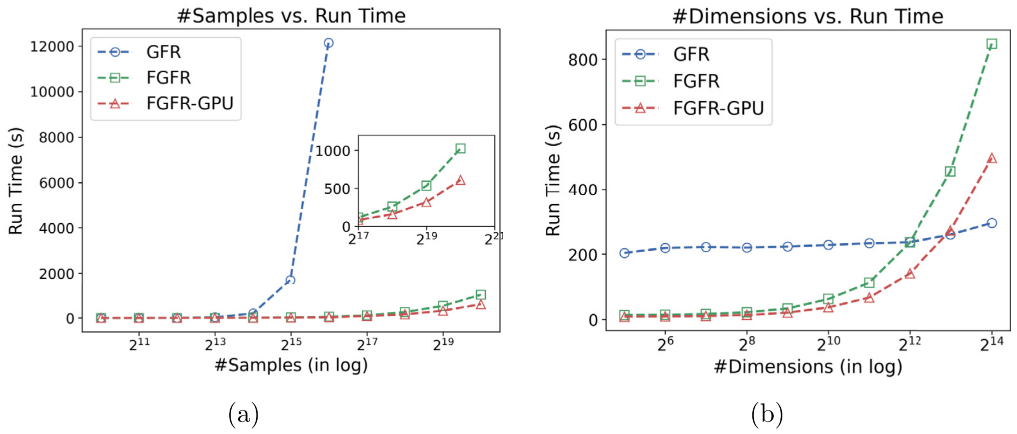

Fig. 10. Running time (in seconds) of the algorithms GFR, FGFR, and FGFR-GPU as a function of (a) the sample number N and (b) the data dimension D (showcased in a semi-log scale). The data dimension and the sample number are fixed at 2^5 and 2^{14} respectively for (a) and (b).

Some interesting observations from Fig. 9 are as follows. A small α can yield good clustering performance over a relatively wider range of iterations for most datasets. The heatmaps on the large datasets (the bottom row of Fig. 9) are relatively smoother than those on the small datasets (the top row of Fig. 9). Moreover, the heatmaps for USPS and MNIST in Fig. 10 are quite similar and smooth over a large area in the α-iteration domain, indicating that a common good choice for α applies to these two datasets. As a rule of thumb, a small α (e.g., 0.025 or 0.05) together with a large iteration number may be a good choice. We omit the heatmaps in terms of AIR since they exhibit similar phenomena as NMI.

5.6. Scalability analysis on N and D

In this subsection, we explore the effects of the sample number N and the data dimension D on the running time of GFR, FGFR, and FGFR-GPU[3] To conduct this analysis, we randomly generate a series of datasets composed of eight Gaussian-like clusters.

For the first experiment, we fix D at 32 and vary N in the range 2^{10} to 2^{20}. The running time of the three algorithms is illustrated in Fig. 10a. Notably, there is no significant difference in the running time of the three methods for smaller datasets ($N < 2^{13}$). However, with increasing dataset size, the running time of GFR escalates exponentially. In contrast, the running times of FGFR and FGFR-GPU remain relatively stable, with FGFR-GPU showing an increasingly substantial efficiency advantage over FGFR as N exceeds 2^{17}. Particularly, when $N = 2^{20}$, FGFR-GPU (~600 s) saves 40% of the running time taken by FGFR (~1000 s).

In the second experiment, we keep N fixed at 2^{14}, while varying D from 2^5 to 2^{14}. As depicted in Fig. 10b, GFR's running time remains fairly stable (around 200 s to 250 s) regardless of the increase in D. This is attributed to the fact that GFR's computational complexity is primarily dominated by the sample number N (as detailed in Table 2), resulting in its insensitivity to D. Conversely, while FGFR and FGFR-GPU initially have running times significantly lower than GFR, they continuously increase as D enlarges. This is mainly due to that FGFR and FGFR-GPU are required to run an additional k-means stage for supporting point extraction, whose computational complexity substantially depends on the data dimension D (refer to Section 4.2).

These findings suggest that FGFR is robust to the increase in data volume given a moderate data dimension, making it suitable for handling datasets with a large number of samples. However, for scenarios where the sample number is small yet the data dimension is extraordinarily large, GFR may be a more appropriate choice. Moreover, the utilization of GPU parallel computing technology can further accelerate FGFR.

6. Conclusion

Based on a new perspective on graph frequency, we have presented GFR, as implemented by Algorithm 1. GFR iteratively enhances the discriminability of potential clusters and the quality of the constructed graph, showcasing remarkable capabilities in overcoming the suboptimality inherent in unsupervised graph construction due to noisy links. Furthermore, a fast version of GFR enables its applicability to large-size datasets. In particular, the proposed GFR can be applied to unsupervised clustering for non-Euclidean data. Extensive experimental results for unsupervised clustering facilitated by the proposed GFR were provided to demonstrate its superior overall clustering performance compared to several state-of-the-art graph-based clustering algorithms. These merits position GFR as an appealing option for researchers and practitioners working with non-Euclidean data, as it offers an effective solution to the challenges posed by unsupervised graph construction. However, it is crucial to acknowledge the potential weaknesses of the GFR method. Similar to other unsupervised methods, the success of GFR relies to some on the clustering assumption of the data distribution, and the method may be less effective when applied to datasets with particularly complex or noisy structures.

[3] It is implemented with a GPU-based parallel computing framework, pyTorch.

Additionally, GFR has been primarily tested on a limited number of real-world datasets, which may not yet comprehensively cover the full spectrum of possible applications. In light of these findings, we suggest several avenues for future work:

1. Develop a stopping criterion to attentively determine the maximum number of iterations, allowing for a more convenient application of GFR.
2. Devise a scheme for the proper choice of the low-frequency component enhancing factor α, which plays a crucial role in controlling the balance between the two types of frequency components to achieve a desired convergence point.
3. Investigate methods for further accelerating GFR, enhancing its robustness to large-scale and real-time applications.
4. Explore the potential applications of GFR in graph neural networks, which can benefit from GFR's ability to improve graph structure and feature discriminability.

CRediT authorship contribution statement

Yangli-ao Geng: Formal analysis, Methodology, Writing – original draft, Writing – review & editing. **Chong-Yung Chi:** Writing – original draft, Writing – review & editing. **Wenju Sun:** Data curation, Software. **Jing Zhang:** Data curation, Software. **Qingyong Li:** Conceptualization, Funding acquisition, Project administration.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgements

This work was supported by the Open Grants of Key Laboratory of Lightning [grant numbers 2023KELL-B002], the Fundamental Research Funds for the Central Universities [grant numbers 2023JBZY037, 2022JBQY007], the National Natural Science Foundation of China [grant numbers 62276019, 62306028, U2034211], the Beijing Natural Science Foundation [grant numbers L231019], the Shanghai Industrial Development Project [grant numbers HCXBCY-2023-033], and the Ministry of Science and Technology [grant numbers MOST 111-2221-E-007-035-MY2].

References

[1] D. Arthur, S. Vassilvitskii, K-means++: the advantages of careful seeding, in: Proc. Annual ACM-SIAM Symposium on Discrete Algorithms, 2007, pp. 1027–1035.
[2] D. Bo, X. Wang, C. Shi, H. Shen, Beyond low-frequency information in graph convolutional networks, in: Proc. AAAI Conference on Artificial Intelligence, vol. 35, 2021, pp. 3950–3957.
[3] C. Bodnar, F. Di Giovanni, B. Chamberlain, P. Liò, M. Bronstein, Neural Sheaf Diffusion: A Topological Perspective on Heterophily and Oversmoothing in gnns, vol. 35, 2022, pp. 18527–18541.
[4] D. Cai, X. Chen, Large scale spectral clustering via landmark-based sparse representation, IEEE Trans. Cybern. 45 (2014) 1669–1680.
[5] E. Chalmers, A.J. Gruber, A. Luczak, Hippocluster: an efficient, hippocampus-inspired algorithm for graph clustering, Inf. Sci. 639 (2023) 118999.
[6] M. Chen, Z. Wei, Z. Huang, B. Ding, Y. Li, Simple and deep graph convolutional networks, in: Proc. International Conference on Machine Learning, 2020, pp. 1725–1735.
[7] S. Chen, Y.C. Eldar, L. Zhao, Graph unrolling networks: interpretable neural networks for graph signal denoising, IEEE Trans. Signal Process. 69 (2021) 3699–3713.
[8] X. Chen, W. Hong, F. Nie, D. He, M. Yang, J.Z. Huang, Spectral clustering of large-scale data by directly solving normalized cut, in: Proc. ACM SIGKDD Conference on Knowledge Discovery & Data Mining, 2018, pp. 1206–1215.
[9] F.R. Chung, C.G. Fan, Spectral Graph Theory (Revised and Improved), American Mathematical Society, 1997.
[10] A. Coates, A. Ng, H. Lee, An analysis of single-layer networks in unsupervised feature learning, in: Proc. International Conference on Artificial Intelligence and Statistics, 2011, pp. 215–223.
[11] Y. Ding, Z. Zhang, X. Zhao, D. Hong, W. Li, W. Cai, Y. Zhan, Af2gnn: graph convolution with adaptive filters and aggregator fusion for hyperspectral image classification, Inf. Sci. 602 (2022) 201–219.
[12] X. Dong, D. Thanou, L. Toni, M. Bronstein, P. Frossard, Graph signal processing for machine learning: a review and new perspectives, IEEE Signal Process. Mag. 37 (2020) 117–127.
[13] Y. Gao, M. Wang, R. Ji, Z. Zha, J. Shen, K-partite graph reinforcement and its application in multimedia information retrieval, Inf. Sci. 194 (2012) 224–239.
[14] D. He, Z. Tang, Q. Chen, Z. Han, D. Zhao, F. Sun, A two-stage deep graph clustering method for identifying the evolutionary patterns of the time series of animation view counts, Inf. Sci. (2023) 119155.
[15] N. Hoang, T. Maehara, T. Murata, Revisiting graph neural networks: graph filtering perspective, in: Proc. IEEE International Conference on Pattern Recognition, 2021, pp. 8376–8383.
[16] R.A. Horn, C.R. Johnson, Matrix Analysis, 2nd ed., Cambridge University Press, New York, NY, USA, 2012.
[17] J. Huang, F. Nie, H. Huang, Spectral rotation versus k-means in spectral clustering, in: Proc. AAAI Conference on Artificial Intelligence, 2013.
[18] P. Huang, X. Yang, Unsupervised feature selection via adaptive graph and dependency score, Pattern Recognit. 127 (2022) 108622.
[19] X. Jiang, Z. Yang, P. Wen, L. Su, Q. Huang, A sparse-motif ensemble graph convolutional network against over-smoothing, in: Proc. International Joint Conference on Artificial Intelligence (IJCAI), 2022, pp. 2094–2100.

[20] D. Jin, Z. Yu, P. Jiao, S. Pan, D. He, J. Wu, P. Yu, W. Zhang, A survey of community detection approaches: from statistical modeling to deep learning, IEEE Trans. Knowl. Data Eng. 35 (2023) 1149–1170.
[21] N. Keriven, Not too little, not too much: a theoretical analysis of graph (over) smoothing, Adv. Neural Inf. Process. Syst. 35 (2022) 2268–2281.
[22] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, in: Proc. International Conference on Learning Representations, 2017.
[23] A. Krizhevsky, Learning Multiple Layers of Features from Tiny Images, Technical Report, Department of Computer Science, University of Torono, 2009.
[24] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proc. IEEE 86 (1998) 2278–2324.
[25] Y. LeCun, O. Matan, B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard, L. Jacket, H.S. Baird, Handwritten zip code recognition with multilayer networks, in: Proc. IEEE International Conference on Pattern Recognition, vol. 2, 1990, pp. 35–40.
[26] B. Lei, Y. Zhu, S. Yu, H. Hu, Y. Xu, G. Yue, T. Wang, C. Zhao, S. Chen, P. Yang, et al., Multi-scale enhanced graph convolutional network for mild cognitive impairment detection, Pattern Recognit. 134 (2023) 109106.
[27] Q. Li, Z. Han, X.-M. Wu, Deeper insights into graph convolutional networks for semi-supervised learning, in: Proc. AAAI Conference on Artificial Intelligence, 2018.
[28] J. Liao, W. Zhou, F. Luo, J. Wen, M. Gao, X. Li, J. Zeng, Sociallgn: light graph convolution network for social recommendation, Inf. Sci. 589 (2022) 595–607.
[29] F. Lin, W.W. Cohen, Power iteration clustering, in: Proc. International Conference on Machine Learning, 2010, pp. 655–662.
[30] W. Liu, J. He, S.-F. Chang, Large graph construction for scalable semi-supervised learning, in: Proc. International Conference on Machine Learning, 2010, pp. 679–686.
[31] U.V. Luxburg, A tutorial on spectral clustering, Stat. Comput. 17 (2007) 395–416.
[32] A. Ng, M.I. Jordan, Y. Weiss, On spectral clustering: analysis and an algorithm, in: Proc. Annual Conference on Neural Information Processing Systems, 2002, pp. 849–856.
[33] A. Ortega, P. Frossard, J. Kovačević, J.M. Moura, P. Vandergheynst, Graph signal processing: overview, challenges, and applications, Proc. IEEE 106 (2018) 808–828.
[34] Y. Rong, W. Huang, T. Xu, J. Huang, Dropedge: towards deep graph convolutional networks on node classification, in: Proc. International Conference on Learning Representations, 2020.
[35] F.S. Samaria, A.C. Harter, Parameterization of a stochastic model for human face identification, in: Proc. IEEE Workshop on Applications of Computer Vision, 1994, pp. 138–142.
[36] K.K. Sharma, A. Seal, Multi-view spectral clustering for uncertain objects, Inf. Sci. 547 (2021) 723–745.
[37] J. Shi, J. Malik, Normalized cuts and image segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 22 (2000) 888–905.
[38] S. Shi, F. Nie, R. Wang, X. Li, Self-weighting multi-view spectral clustering based on nuclear norm, Pattern Recognit. 124 (2022) 108429.
[39] G. Sun, Y. Cong, J. Dong, Y. Liu, Z. Ding, H. Yu, What and how: generalized lifelong spectral clustering via dual memory, IEEE Trans. Pattern Anal. Mach. Intell. 44 (2021) 3895–3908.
[40] H. Tao, J. Qiu, Y. Chen, V. Stojanovic, L. Cheng, Unsupervised cross-domain rolling bearing fault diagnosis based on time-frequency information fusion, J. Franklin Inst. 360 (2023) 1454–1477.
[41] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, Y. Bengio, Graph attention networks, in: Proc. International Conference on Learning Representations, 2018.
[42] L. Wu, H. Lin, B. Hu, C. Tan, Z. Gao, Z. Liu, S.Z. Li, Beyond homophily and homogeneity assumption: relation-based frequency adaptive graph neural networks, IEEE Trans. Neural Netw. Learn. Syst. (2023) 1–13.
[43] H. Xiao, K. Rasul, R. Vollgraf, Fashion-MNIST: A Novel Image Dataset for Benchmarking Machine Learning Algorithms, 2017.
[44] W. Ye, S. Goebl, C. Plant, C. Böhm, Fuse: full spectral clustering, in: Proc. ACM SIGKDD Conference on Knowledge Discovery & Data Mining, 2016, pp. 1985–1994.
[45] J. Yin, S. Sun, Incomplete multi-view clustering with reconstructed views, IEEE Trans. Knowl. Data Eng. 35 (2023) 2671–2682.
[46] Z. Zhang, P. Cui, X. Wang, J. Pei, X. Yao, W. Zhu, Arbitrary-order proximity preserved network embedding, in: Proc. ACM SIGKDD Conference on Knowledge Discovery & Data Mining, 2018, pp. 2778–2786.
[47] L. Zhao, L. Akoglu, Pairnorm: tackling oversmoothing in gnns, in: Proc. International Conference on Learning Representations, 2020.
[48] Z. Zhuang, H. Tao, Y. Chen, V. Stojanovic, W. Paszke, Iterative learning control for repetitive tasks with randomly varying trial lengths using successive projection, Int. J. Adapt. Control Signal Process. 36 (2022) 1196–1215.
[49] Z. Zhuang, H. Tao, Y. Chen, V. Stojanovic, W. Paszke, An optimal iterative learning control approach for linear systems with nonuniform trial lengths under input constraints, IEEE Trans. Syst. Man Cybern. Syst. 53 (2023) 3461–3473.